LISTing Newsletter

March 1994

Newsletter of the Long Island Sinclair/Timex Users Group

Next Meeting
March 13

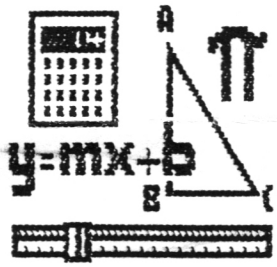


TABLE OF CONTENTS

Listing Policy

Annual Dues $16.00

One "sample" copy sent upon receipt of Business size SASE. Copies provided on EXCHANGE BASIS with other bona fide user groups. LISTing is published monthly except July and August by LIST (Long Island Sinclair Timex) Group, a not for profit user group.

We are always looking for articles, programs, reviews etc. to keep our members informed and entertained. You maintain full credit and copyright.

Portions if this publication may be reproduced, without written consent. Please give credit to LIST when reprinting articles.

LIST disclaims any responsibility for any damage you may do to your computer as a result of reading any articles in LISTing.

LIST OFFICERS

PRES. HARVEY RAIT
V.P. BOB GILDER
TRES. ROBERT MALLOY
COR.SEC. JOHN PAZMINO
EDITOR. FRED STERN
LIBR. TOM SKAPINSKI

PLEASE SEND INQUIRIES TO:
LIST
MR. HARVEY RAIT
3 PERI LANE
VALLEY STREAM, N.Y. 11581

PLEASE SEND SUBMISSIONS TO:
LISTING
MR. FREDERIC STERN
P.O. BOX 264
HOLBROOK, N.Y. 11741

COMING EVENTS:

MAR. 13, 1994 LIST MEETING.

SPECIAL NOTICE

THE NEXT MEETING WILL BE HELD AT THE ICE CREAM DISPENSARY (HARVEYS STORE)
334 DOGWOOD AVENUE
FRANKLIN SQUARE N.Y.
TEL: 516-486-1090

DIRECTIONS: SOUTHERN STATE PKWY TO EXIT 17 NORTH HEMPSTEAD AVE. GO TO FIRST TRAFFIC LIGHT, LEFT TURN ON TO CORNWALL, NEXT TRAFFIC LIGHT, BEAR RIGHT ON TO DOGWOOD AVENUE. GO 1 MILE TO THE ICE CREAM DISPENSARY, IN A SMALL SHOPPING CENTER ON THE LEFT SIDE OF THE ROAD.

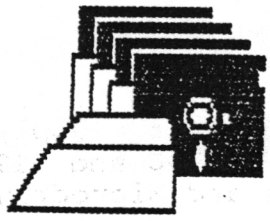
MEETING MINUTES
REPORTED BY: FRED STERN
FEB. 13, 1994

THE MEETING WAS CALLED TO ORDER BY HARVEY AT 2:30PM

IN THE MAIL WE RECEIVED 4 MEMBERSHIP RENEWALS

WHEN YOU SEND IN YOUR RENEWAL, PLEASE LET US KNOW WHAT SYSTEM YOU ARE USING. IE, TS1000, QL TS2068, ETC...

KEN (MR. ULTRALIGHT) LANG COULD NOT MAKE THE MEETING. WE HOPE HE WAS ABLE TO SHOVEL THE SNOW-DRIFT OFF OF HIS CAR.



SURPRISE SALE

THIS IS AN EXAMPLE OF WHY YOU CAN NOT MISS A LIST MEETING. AFTER A LONG ABSENCE, WE WERE PLEASED TO WELCOME BACK A LONG-TIME LIST MEMBER, MR. EDGAR GROSS.
EDGAR, A COMPUTER PROGRAMER BY TRADE, BOUGHT AN IBM PC WHICH HE NEEDS FOR HIS BUSINESS. SO, HE BROUGHT HIS TIMEX EQUIPMENT TO SELL TO HIS FELLOW LIST MEMBERS.
BOB MALLOY PICKED UP A QL, RGB MONITOR AND QL SOFTWARE FOR $150.00. MICHAEL AND FRED STERN BOUGHT A TS2068, AERCO DISK DRIVE INTERFACE, AERCO PRINTER INTERFACE AND TS2068 SOFTWARE FOR $50.00.
EVERYONE ENJOYED THIS SURPRISE SALE, BOTH SELLER AND BUYERS WENT HOME HAPPY.
WE ALL WISH EDGAR MUCH HAPPINESS ON HIS RECENT MARRIAGE AND MUCH SUCCESS IN HIS FUTURE BUSINESS ENDEAVORS.

THE BALANCE OF THE MEETING WAS A TECHNIDAL ROUNDTABLE.

CLASSIFIEDS

THIS CLASSIFIED SECTION IS AVAILABLE TO ALL LIST MEMBERS FREE OF CHARGE.
THE ONLY RESTRICTION IS THAT IT IS TO BE USED ONLY FOR THE SEEKING, SELLING OR SWAPPING OF SINCLAIR, TIMEX OR MICROACE COMPUTER EQUIPMENT, PERIPHERALS AND SOFTWARE.
LISTING, LIST, AND ITS OFFICERS DO NOT ENDORSE, WARRANTY, OR GUARANTEE ANY OF THE ITEMS LISTED IN THIS CLASSIFIED SECTION

THE FOLLOWING PUBLICATIONS ARE AVAILABLE ONLY THROUGH LIST:

ZX-81/TS1000 TECHNICAL TIDBITS
TECHNICAL TIDBITS PART II
SAVINGS AND LOAD OF THE TIMEX COMPUTER
$4.00 EACH.

I AM LOOKING FOR AN AERCO DISK DRIVE INTERFACE FOR THE TS1000. I WILL CONSIDER A PURCHASE EITHER WITH OR WITHOUT DRIVES. I WILL EVEN CONSIDER A U-REPAIR. FRED STERN 516-737-0983. EVENINGS AND WEEKENDS.

CONTINUE PAGE 4

QL CORNER

It seems that I had left out two important voltage readings pertaining to the QL Power Supply article last month. Both voltages supply AC voltage to a +12VDC and -12VDC Voltage regulators, which are used for the Serial ports and Microdrives.

Looking at the printed circuit board layout (foil side), you will notice that three lands are marked 'R', 'G' and 'B'. These pads are where the power cable connects to the power supply and the R G B leters printed on these pads represent the color of the three cables nested within the power cable. Red is soldered to R and so on.

Set your Volt meter at least to 20 Volts AC. Plug the power supply AC connector into an AC outlet. Place the Black meter lead on 'Blue" and the Red meter lead on "RED'. You should find a reading of approximately 17.30 VAC. If you are using a Digital volt meter you can leave the black lead of your meter on blue and place the Red meter lead on the"Green" pad. You should get a similar voltage reading as before. If you are using an Analog volt meter switch the leads as stated for the digital volt meter.

As I find additional information which would be necessary for trouble shooting the QL power supply, it will appear in "The QL Corner.

A new piece of QL software has surfaced last month from Deltasoft which is titled "Flightdeck - 1.03". Flightdeck allows the user to fly a twin-engine jet airliner and provides 3-D views of the 'world' outside. A three and a half page review for this program is reviewed in the December issue of QL World magazine. Flightdeck-1.03 can be purchased from Dilwyn Jones Computing, UK. DJC's address appears in several past issues of LISTing and sports a price of £15.00. Incidentally, it is my understanding that Dilwyn Jones will be attending the QL NEWPORT show this coming May.

Qubbesoft (UK) is launching a new IDE hard disk interface, named QUBIDE. Stuart of Miracle Systems provided the way to connect the interface to the QL ROM port at the back of the QL. The price will range from £75.00 to £100.00. QUBIDE will drive any hard disk up to 120 Megabytes. When additional information surfaces, you will be informed in QL Corner.

While the subject of QL Hard Disk interfaces; a DIY approach has become available by German author Dirk Steinkopks. Dirk has developed a small hardware circuit with only four TTL gates which connects to the QL expansion connector and sends the proper signals to an IBM PC/XT slot for connection to a Hard Disk board. This interface has been tested with OMTI and Western Digital Hard Disk controllers and suits both MFM and RLL drives.

Software and schematic of Dirk's interface is available from Qubbesoft as a Public Domain disk, Special 22. The software is written in both English and German. It has been stated that this interface performs happily with either a Gold Card or Trump Card. Perhaps the software will show up at the Miracle in Newport II show this May. If so, I will advise our readers of its availability. This information stems from the latest issue of QL World Magazine.

See you naxt month.....Bob Gilder

QL CORNER EXTRA

February 12th 1994

I have just received information from Bob Dyl, Editor of IQLR (International QL Report), that Miracle Systems LTD, UK has introduced a Super Gold Card at a Computer show in the UK. This new interface operates with a 68020 Microprocessor running at 25 Mhz, comes with Four Megs of RAM on board, has a four drive adapter and a "true" Parallel Printer interface built in.

The price is in the range of £325.00. Miracle will take a trade in for an Issue 1, 2, or 3 Gold Card allowing approximately £150.00. I am not sure that I have the prices right at this time, however, it should come close to the actual cash value. I will have all the information for the next issue of LIST. Perhaps, they will sell the Super Gold Card with less memory as the price of memory chips in the UK are much higher than in the USA. We could add the additional memory upon receipt of the interface.

Miracle will no doubt sell their new interface at the "Miracle in Newport II" show in Rhode Island this spring, May 14th. I intend to bring an issue 3 Gold Card at this show for a trade up.

Bob Gilder

A FINAL WORD

MY NAME IS FEEL STEER AND I AM THE EDITOR OF THIS EDITION OF LISTING.

I WOULD LIKE TO THANK ALL OF YOU WHO ANSWERED ME AND SENT THE DOCUMENTATION I NEEDED FOR THE MINIMOD 1.8 AND I-COMM. PROGRAMS. I WILL HAVE PROGRAMS AND DOCUMENTATION WHICH I PROMISED IN RECIPROCATION OUT TO ALL OF YOU VERY SOON.

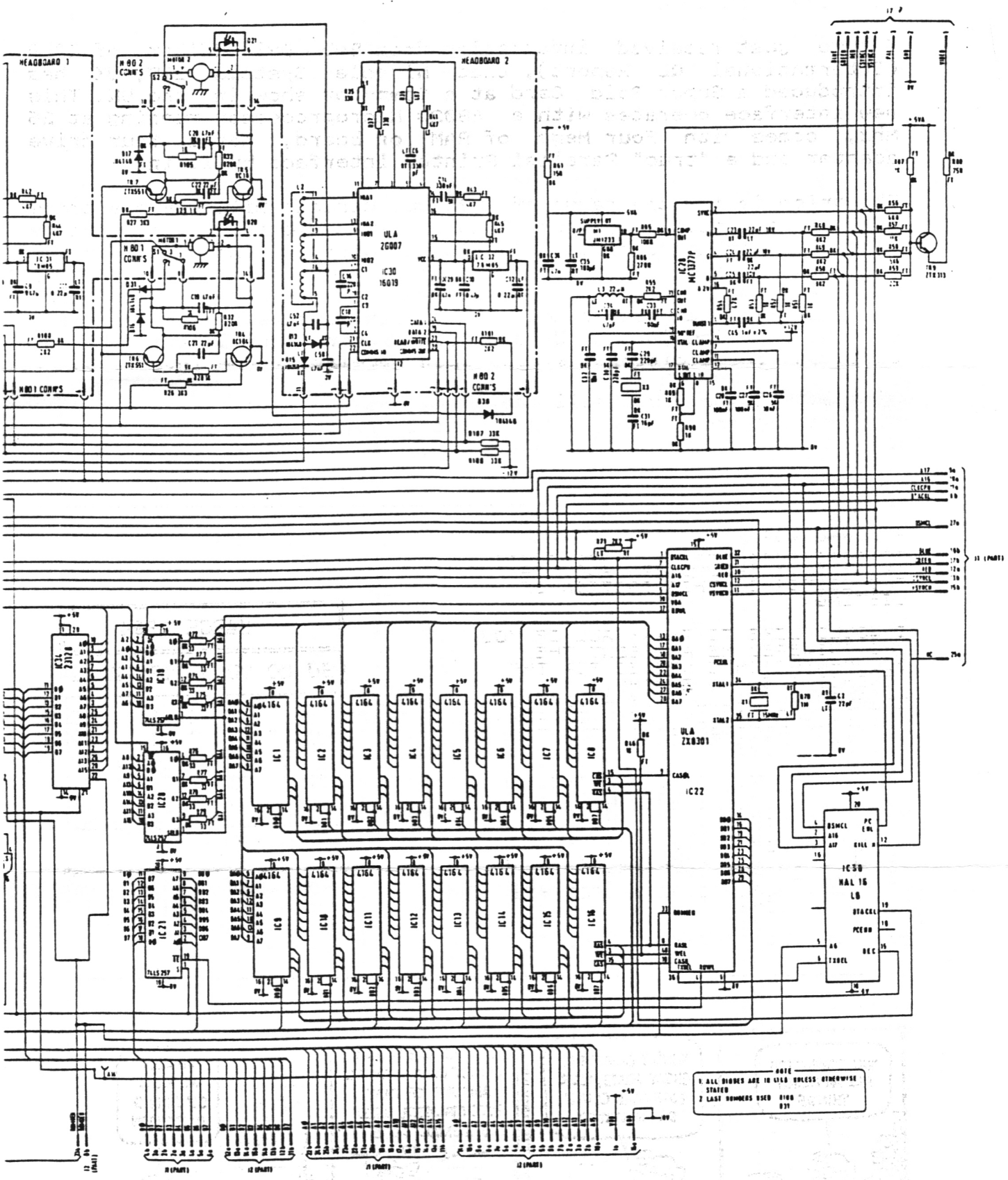
THANK YOUS TO TOM SKAPINSKI, AND BOB GILDER FOR THERE HELP AND CONTRIBUTIONS TO THIS ISSUE.

A VERY SPECIAL THANK YOU TO HARVEY FOR HIS HOSPITALITY, AND THE USE OF HIS STORE FOR OUR MEETING. ALSO TO MIKEY FOR HIS CONTRIBUTIONS.

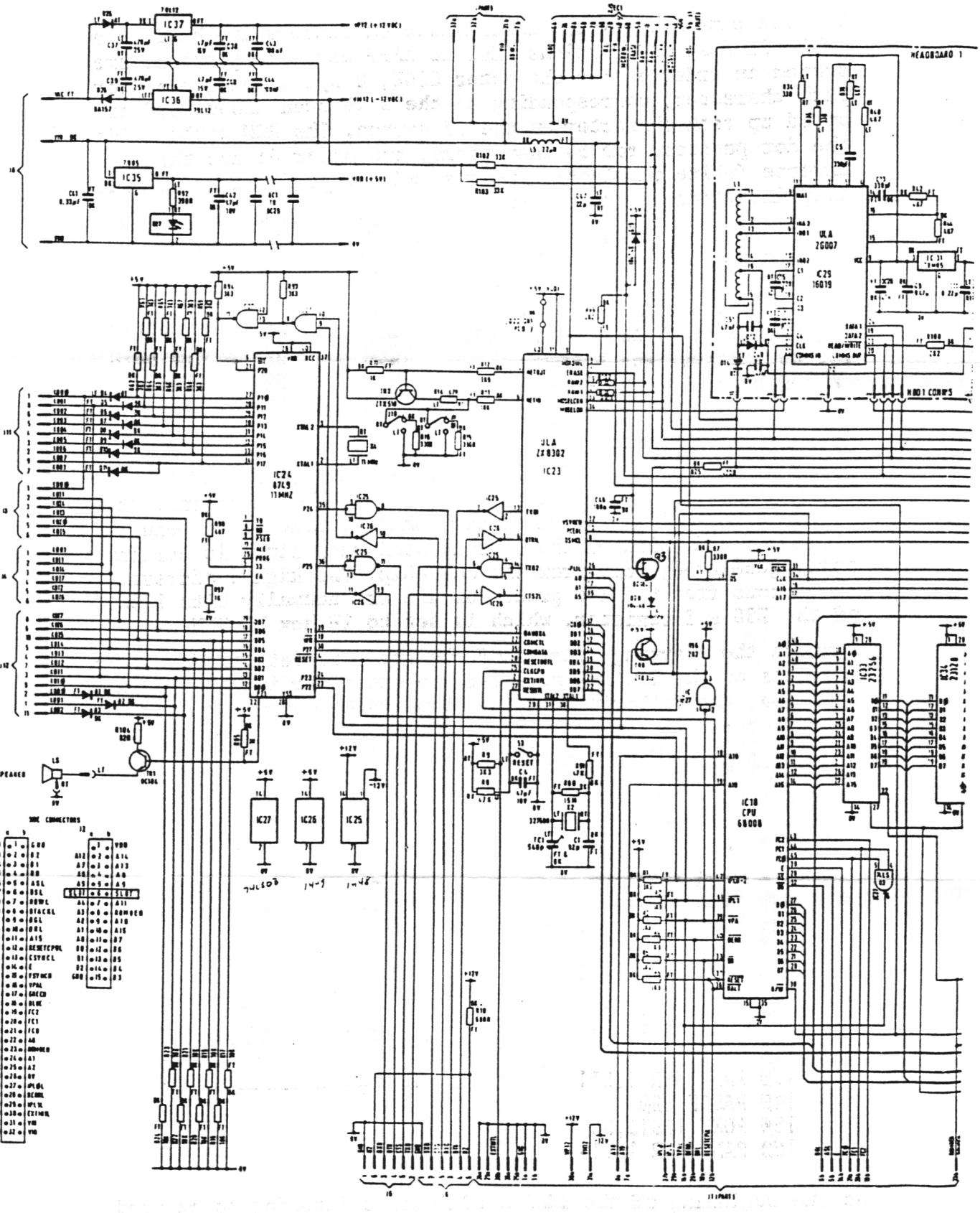
SEE YOU ALL AT THE NEXT MEETING.

MARCH 1994						
SU	MO	TU	WE	TH	FR	SA
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		
LIST MEETING 13TH						

BETTY



Q.L. CIRCUIT DIAGRAM PART I



Q.L. CIRCUIT DIAGRAM PART II

Pseudo-ROM

The dot patterns for the displayable characters are held in a table in ROM, in locations 1E00 to 1FFF hex. The patterns are stored in order of the character CODE, 8 bytes being used for each character, corresponding to the 8 TV scan lines used to build up each character on the TV screen. The ROM table holds the dot patterns for 64 characters, the other 64 are the inverse (white on black) versions and are produced by the SCL chip inverting the video signal.

Looking at, for example, the character '0', this has the CODE 28 (1C hex) and is generated by the 8 ROM bytes starting at 1EE0 hex (1E00 + 1C x 8). The hexadecimal and binary values of these 8 bytes are shown alongside and the pattern can be seen in the binary version.

Addr.	Hex.	Binary
1EE0	00	00000000
1EE1	3C	00111100
1EE2	46	01000110
1EE3	4A	01001010
1EE4	52	01010010
1EE5	62	01100010
1EE6	3C	00111100
1EE7	00	00000000

When the ZX81 is producing the display, it presents 13 bit addresses to the ROM to select the wanted 8 bit patterns. The low 9 address lines, which specify which character is being displayed and which of the 8 horizontal scan lines is currently being generated, come from the SCL chip. The high 4 address lines come from the Z80 processor and are actually bits 1 to 4 of the Z80's I register, which is set to 1E hex by NEW.

Because the starting address of the table is determined by the contents of the Z80's I register, we can alter it. As a demonstration, the following program displays 64 characters then loads the I register with 0, making the ZX81 use the beginning of the ROM as a dot pattern table. Ten seconds later, the display will be returned to normal by restoring the correct value to the I register.

```
  1 REM 12345
 10 POKE 16514,62
 20 POKE 16515,0
 30 POKE 16516,237
 40 POKE 16517,71
 50 POKE 16518,201
100 FOR A=0 TO 63
110 PRINT CHR$ A;
120 NEXT A
130 RAND USR 16514
140 PAUSE 500
150 POKE 16515,30
160 RAND USR 16514
```

The machine code routine used is;

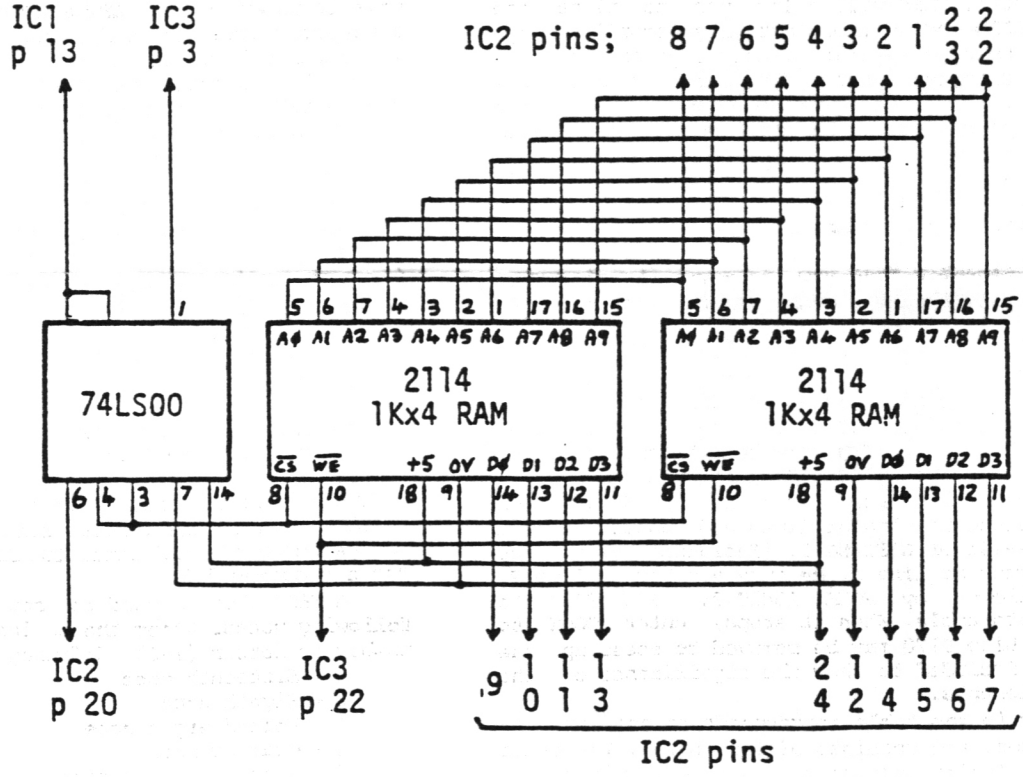
```
3E 00     LDA 0
ED 47     LD I A
C9        RET
```

As the beginning of the ZX81's ROM wasn't intended to be used as a dot pattern table, it gives a meaningless display ! But, if we were to add some RAM to the ZX81, in such a way that the

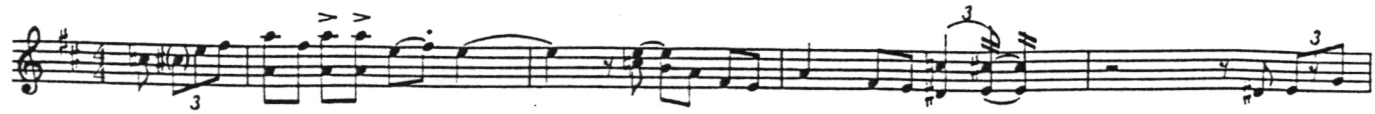
display generation circuits could use it, then we could POKE any dot patterns we wanted into this 'Pseudo ROM', and so create our own character set. This could include shapes for games programs, mathematical or engineering symbols, or elements for high resolution drawings.

The circuit given below shows how 1K of RAM can be added in this way. Because all of the wanted signals are not present on the ZX81's rear connector, the 'Pseudo ROM' is best fitted inside the ZX81's case, with the connections wired directly as shown. The extra current taken from the +5V supply will make the ZX81's regulator run slightly hotter, but you have drilled cooling holes in the case, haven't you ?

It occupies addresses 2000-23FF hex (8192-9215 decimal). This is enough for 2 sets of 64 characters, or you could load just one set of characters and have 512 bytes left to hold machine code or other data. The routine given on the previous page can be used to force the ZX81 to use the dot patterns held in the Pseudo ROM by changing the value POKEd by line 20 to 32 to select the lower ½K of the Pseudo ROM, or 34 to select the upper half. Data (including dot patterns) stored in the Pseudo ROM won't be SAVEd, but neither is it affected by NEW or LOAD, so you can run one program to fill the Pseudo ROM with appropriate data, then LOAD another program to use it.



Note; R28 (680 ohm) to be removed from ZX81 board.



FROM: TIME DESIGNS

MoTSart

Super Music for the ZX81/TS1000/TS1500

Zack Xavier Haquer

The ZX81-type computers don't have "sound", right? WRONG! If you ever listened to a tape of a computer program, you'll realize that it is capable of screechy noises that could loosely be called "sound". Well, ok, but it isn't capable of music, right? WRONG AGAIN! If you've envied those other machines that can beep out a tune or a laser sound, there's no longer any need to feel left out.

The machine-code program presented here gives you three octaves of sound. As listed, the lowest note is the "A" just below "Middle C", but you can move your spectrum up or down as desired. But that's not all; it is easy to write and play music using only BASIC commands, thanks to a built-in "music interpreter". No, you can't do multiple voices (though you can simulate two voices as shown in the demo), and you can't vary the envelope or modify the waveform. Also, since it has to run in FAST mode, you can't see your display screen while the music is playing (unless you have the Oliger video upgrade). Still, this relocatable machine code routine might be just what the conductor ordered to tune in your BASIC software.

You won't need much in the way of hardware. If you wish, you could connect a mic-level amplifier/speaker to the MIC output. Alternately, simply use your cassette deck. Connect the MIC from the computer to the MIC jack on the recorder, and connect an earphone or a small speaker to the EAR jack of the recorder. Then take an old useless cassette, and cut or remove the tape, making a "dummy" cassette. This will allow you to place the recorder in RECORD mode without actually recording anything. Or you could, of course, simply save to tape, and listen to your computer sonata after it has been recorded. Finally, a cheap AM radio located near the computer might pick up the sound, but with reduced quality.

The machine-code routine takes up exactly 256 bytes. It is fully relocatable, so you can place it anywhere you want. A good place is in a 1 REM statement; this article will assume that this is where you'll want to put it, but remember that you can move it elsewhere if you wish. Simply change the LET BEEP= statement to match the start of your code.

Also required is 144 bytes after the code, for the frequency/duration lookup table. So start by entering a 1 REM followed by 400 X's or other character. Use a POKER program such as LISTING 2 of the "Kaleidoscope" article in TDM Vol.3 No.1 You only have to change line 3 to read: FOR A=16514 TO 16769. RUN your loader program, and enter the decimal values given in TABLE 1. When you're done, delete the loader lines and enter line 2 and lines 9000-9991 of LISTING 1. (Incidentally, if your listing gets stuck at line 1, be sure you have a line 2, then LIST 2 followed by POKE 16419,2.) RUN 9000 to generate the data table. When it stops, enter CONT to fill the rest. Line 9170 may be removed to speed up the process; it is included to show the significance of the various data elements.

Each entry in the table corresponds to one of the 36 possible notes, and consists of four bytes. The first two give the "delay constant" that determines the frequency of the note. (We call this "BC" in the program since this is the register pair used for this purpose.) The second two bytes give the number of cycles required for each note, at the minimum possible duration.

The signal generated by the routine is perfectly symmetrical (50% duty cycle). The minimum ON and OFF time (BC=1) is 199 "T" states, and each increment of BC increases this time by 26 T states. Though the actual clock frequency is 3.5 mHz., the "effective" clock frequency of the machine is 3.192 mHz. (T-states per second) because of the keyboard-sensing routine in the non-maskable interrupt. So, the frequency outputted will be:

FREQ=3.192E6/(2*(199+26*(BC-1)))

By transposing this equation, the value of BC for a given frequency is:

BC=1 + ((1.596E6/FREQ)-199)/26

Each note (half-step) will be a fixed ratio higher in frequency than the previous one. Since there are 12 half-steps per octave, and each octave represents a doubling in frequency, this ratio is 2^(1/12). The program calculates each frequency using this ratio, and prints is as the first entry in the screen table. From this is calculates BC, and then "back-calculates" the ACTUAL frequency which will result. (Since only integer values can be POKEd, there will always be some imprecision in the actual frequency.) Finally, it calculates how many cycles of each note are required for the minimum time interval (sixteenth note at the fastest tempo).

The "A" below Middle C is defined at 440 Hz. You can move your scales up or down by changing line 9040. For example, to move it down an octave LET FREQ=220, to move it up an octaye, LET FREQ=880, etc. You can even transpose music to different scales by using other values. For instance, if you wrote a piece in the key of C and wish to transpose it to E (4 half-steps higher), simply define your lowest "A" to C♯ (LET A=554.4).

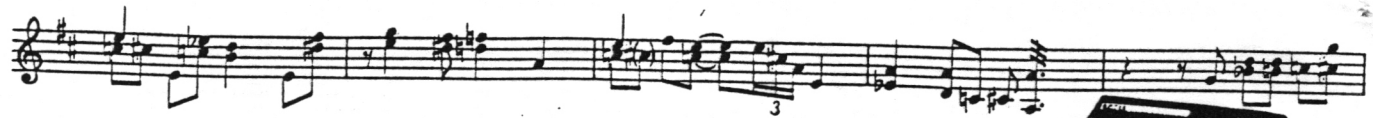
When experimenting with different ranges, you should be aware that the lower notes, the more accurate the PITCH becomes. On the other hand, the higher notes, the more accurate the DURATION becomes.

After generating your table, enter the rest of LISTING 1. Then RUN for a demo. Enter the desired tempo (more about that later). A good tempo for the first music demo (line 100) is 180. Press any key when done for the second part (line 200) which gives a "laser" effect. Again press any key for a "siren" demo (line 300). Finally, press a key to play the second music demo (line 400), which show how you can simulate two voices! A suggested tempo for this is 240. Pretty neat, eh?

Here's how "MoTSart" is used. Your BEEP command must always be of the form: IF USR BEEP THEN... where BEEP has been initialized to the start of the program (16514 in this case). What follows the REM are your musical commands and data. COMMANDS include semicolons ";" and commas ",".

A SEMICOLON is used to set the duration of the following notes. After the semicolon command must be a number or letter (1-G), defining duration as follows:

1 - Sixteenth note
2 - Eighth note
3 - Dotted eight note
4 - Quarter note
6 - Dotted quarter note
8 - Half note
C - Dotted half note
G - Whole note



Values inbetween will give other (unorthodox) note durations. Once the duration has been set, it will remain in force until changed by another semicolon command.

COMMAS are used to tell MoTSart that a note or rest is to be played. If the comma is followed by a single space, a rest is played. Otherwise, you must follow the comma with a number (1-3) specifiying the octave, followed by a letter (A-G) specifying the note (pitch). The note letter may be followed by "accidentals", represented as "+" for sharps and "-" for flats. Note that you can't flat the lowest A (1A), or sharp the highest G (3G).

The only other command is to set the overall TEMPO. This is done using the RAND command. See the demo; simply RAND 1200/(desired tempo). The number you divide into 1200 represents the number of beats (quarter notes) per minute; 120 represents two quarter-notes per second, or one 4/4 bar every two seconds.

MoTSart generates its own error codes. Error R means that you are trying to use a non-valid way of calling the routine, or don't have a REM after the IF USR BEEP THEN. Error (inverse semicolon) means that a SEMICOLON command (duration) is out of range. Finally, all other errors are trapped with error (inverse comma), which means that a COMMAN command is incorrectly formatted or out of range. If you get a semicolon or comma error report, the offending character in the line is flagged by turning to inverse video. BEWARE of errors right at the end of the line! If this happens, the end-of-line marker gets POKEd out, causing the next line to be "strung" together with your BEEP line. If you're not careful and manage to fall into this trap, DO NOT try to edit the line! The best thing to do in this case is to delete the line and re-enter it from scratch.

Now you and your ZX/TS can make beautiful music together!

TABLE 1: MOTSART DECIMAL DATA

33	0	1	9	34	118	64	205
35	15	42	22	64	35	126	254
234	40	2	207	26	35	126	254
25	32	20	35	126	214	29	254
16	56	6	126	203	255	119	207
124	60	50	9	64	24	230	254
118	200	254	26	40	6	126	203
255	119	207	125	35	126	167	32
24	58	9	64	95	58	50	64
87	1	45	6	11	120	177	32
251	21	32	245	29	32	238	24
188	214	29	254	3	48	215	245
35	126	214	38	254	7	48	206
167	40	23	254	3	48	3	60
24	16	32	4	198	2	24	10
254	6	40	4	198	3	24	2
198	4	79	35	126	254	21	32
3	12	24	8	254	22	32	3
13	24	1	43	241	71	128	128
7	7	129	254	255	40	151	254
54	40	147	79	6	0	229	42
118	64	9	9	9	9	78	35
70	35	94	35	86	225	24	2
24	157	175	245	175	245	197	211
255	11	120	177	32	251	245	0
62	0	62	7	61	32	253	241
193	197	219	254	11	120	177	32
251	193	241	60	253	190	9	40
11	35	43	245	62	3	61	32
253	241	24	209	241	60	253	190
50	40	6	221	9	0	0	24
194	27	122	179	32	188	24	184

LISTING 1: BASIC

```
  2 REM MOtsART
  3 GOTO 100
 10 REM SET TEMPO SUBROUTINE
 20 PRINT "TEMPO?"
 30 INPUT TEMPO
 40 RAND 1200/TEMPO
 50 LET BEEP=16514
 60 RETURN
 80 REM WAIT SUBROUTINE
 90 IF INKEY$="" THEN GOTO 90
 95 RETURN
100 REM MUSIC DEMO
105 GOSUB 10
110 IF USR BEEP THEN REM ;8,2E;
5,2G;1, ;2,2G;C,2C;4,2D,2E,2F,2G
,3A;G,2D;8,2E;5,2F+;1, ;2,2F+;C,
2G;4,3A;3,3B;1,2B;4,3B;3,3A;1,2A
;4,3A;C,2G;2,2D,2E;6,2F;2,2E;4,2
D;2,2E,2F;6,2G;2,2F;4,2E;2,2F,2G
;4,3A,2G,2F,2E;8,2D;1,1D;2,2D,2E
;6,2F;2,2E;4,2D;2,2E,2F;6,2G;2,2
F;4,2E,2C,2D,2G;2,2G,2F+,2E,2F+;
G,2G
120 IF USR BEEP THEN REM ;8,2E;
5,2G;1,1G;2,2G;G,2C;6,2F;2,2F;6,
3A;2,3A;G,2D;8,2G;6,2G+;2,2G+;4,
3A,2F,2E,2D;8,2C,2D;G,2E;8,2G;6,
3C;2,3E;4,3A,2F,2E,2D;8,2G,2B;G,
2C;2,2C,2E,2G,3C,3A,2F,2D,2E-,2E
,2G,3A,3B,3C;3,2C,1C
130 GOSUB 80
140 CLS
150 PRINT "RUN AGAIN?"
160 PAUSE 4E4
170 IF INKEY$="Y" THEN GOTO 100
200 REM LASER
210 LET TEMPO=1200
220 GOSUB 40
230 FOR N=1 TO 10
240 IF USR BEEP THEN REM ;1,3G+
,3G,3F+,3F,3E,3E-,3D,3C+,3C,3B,3
B-,3A,2G+,2G,2G-,2F,2E,2E-,2D,2C
+,2C,2B,2B-,2A,1G+,1G,1G-,1F,1E,
1E-,1D,1C+,1C,1B,1B-,1A
245 IF USR BEEP THEN REM ;G,
250 NEXT N
260 GOSUB 80
300 REM SIREN
310 LET TEMPO=600
320 GOSUB 40
330 FOR N=1 TO 5
340 IF USR BEEP THEN REM ;1,1A,
1A+,1B,1C,1C+,1D,1D+,1E,1F,1F+,1
G,1G+,2A,2A+,2B,2C,2C+,2D,2D+,2E
,2F,2F+,2G,2G+,3A,3A+,3B,3C,3C+,
3D,3D+,3E,3F,3F+,3G,3G+
350 IF USR BEEP THEN REM ,3G+,3
G,3F+,3F,3E,3E-,3D,3C+,3C,3B,3B-
,3A,2G+,2G,2G-,2F,2E,2E-,2D,2C+,
2C,2B,2B-,2A,1G+,1G,1G-,1F,1E,1E
-,1D,1C+,1C,1B,1B-,1A
360 NEXT N
370 GOSUB 80
400 REM SIMULATING 2 VOICES
410 GOSUB 10
420 GOSUB 470
430 GOSUB 480
440 GOSUB 480
450 GOSUB 470
460 IF USR BEEP THEN REM ,2C,1E
,2C,1E,2C,1E,2C,1E;4,2C
465 GOSUB 80
466 STOP
470 IF USR BEEP THEN REM ;1,2C,
1E,2C, ,2C,1E,2C, ,2G,2E,2G, ,2G
,2E,2G, ,3A,2F,3A, ,3A,2F,3A, ,2
G,2E,2G,2E,2G,2E,2G, ,2F,2D,2F,
,2F,2B,2F, ,2E,2C,2E, ,2E,1G,2E,
,2D,2A,2D, ,2D,1F,2D, ,2C,1E,2C
,1E,2C,1G,2C,1G
475 RETURN
480 IF USR BEEP THEN REM ,2G,2C
,2G, ,2G,2E,2G, ,2F,2D,2F, ,2F,2
B,2F, ,2E,2C,2E, ,2E,1G,2E, ,2D,
2A,2D,2A,2D,2B,2D, ;1
485 RETURN
9000 REM SET UP DATA TABLE
9010 CLS
9020 PRINT "FREQ   BC  ACTUAL NO
.CYCLES
9030 LET BYTE=16770
9040 LET FREQ=440
9050 LET RATIO=2**(1/12)
9060 FOR N=1 TO 36
9070 LET BC=INT (((1.596E6/FREQ)
-199)/26+1.5)
9080 IF BC=0 THEN LET BC=1
9090 LET ACT=1.596E6/(199+26*(BC
-1))
9100 LET DE=INT (ACT*.013+.5)
9110 RAND BC
9120 POKE BYTE+0,PEEK 16434
9130 POKE BYTE+1,PEEK 16435
9140 RAND DE
9150 POKE BYTE+2,PEEK 16434
9160 POKE BYTE+3,PEEK 16435
9170 PRINT (INT (FREQ*10+.5))/10
;TAB 7;BC;TAB 11;(INT (ACT*10+.5
))/10;TAB 18;DE
9180 LET BYTE=BYTE+4
9190 LET FREQ=FREQ*RATIO
9200 NEXT N
9210 STOP
9990 SAVE "MOTSARt"
9991 GOTO 100
```

Note: line 350 is the same as line 240.

Line 245 ends with a space.